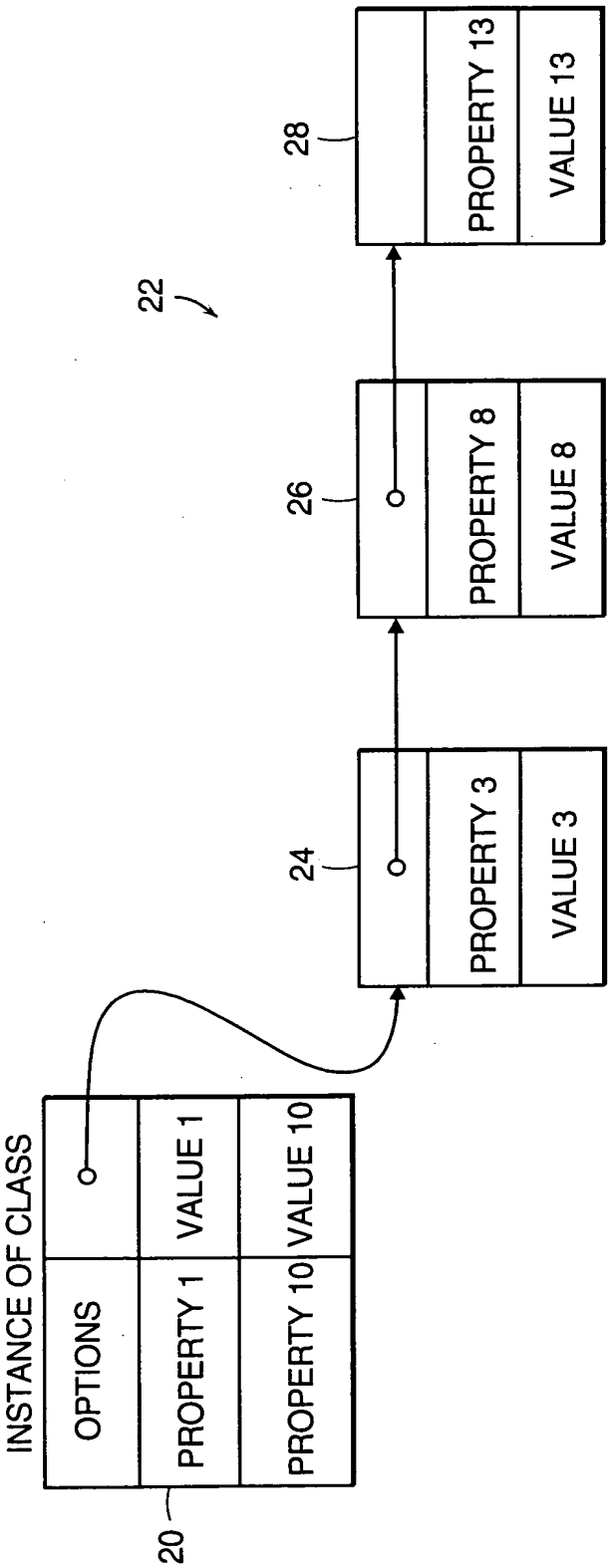

FIG. 1

FOSTER 2696260

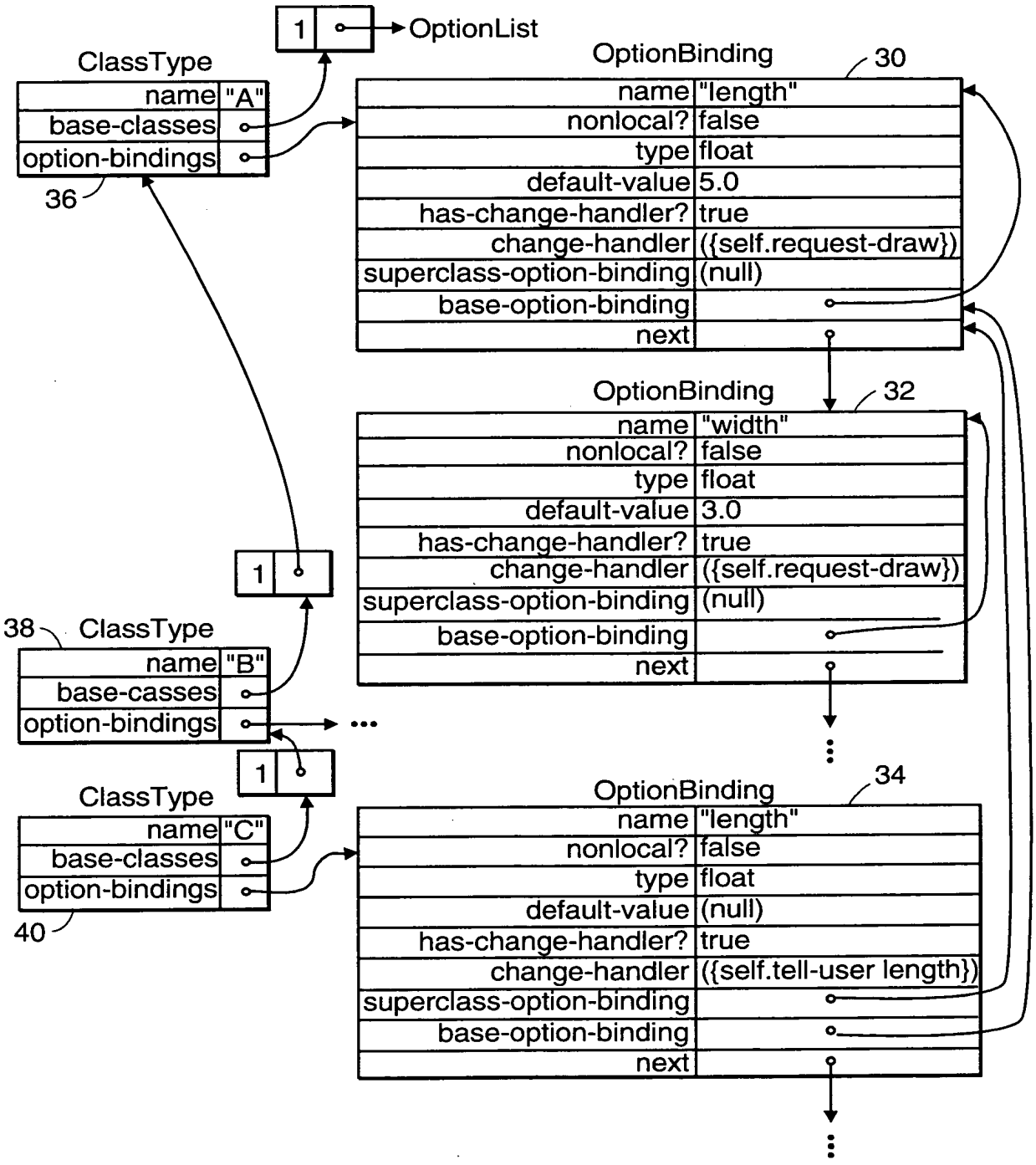
1
OptionList
ClassType
name "A"
base-classes
option-bindings
36
OptionBinding
30
name "length"
nonlocal? false
type float
default-value 5.0
has-change-handler? true
change-handler ({self.request-draw})
superclass-option-binding (null)
base-option-binding
next
OptionBinding
32
name "width"
nonlocal? false
type float
default-value 3.0
has-change-handler? true
change-handler ({self.request-draw})
superclass-option-binding (null)
base-option-binding
next
1
38
ClassType
name "B"
base-casses
option-bindings
1
ClassType
name "C"
base-classes
option-bindings
40
OptionBinding
34
name "length"
nonlocal? false
type float
default-value (null)
has-change-handler? true
change-handler ({self.tell-user length})
superclass-option-binding
base-option-binding
next

FIG. 2A

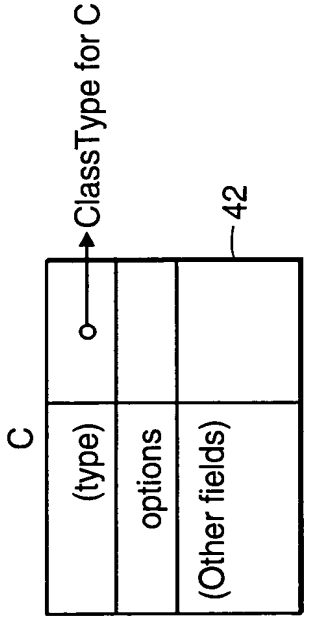
C
(type)
options
(Other fields)
42
ClassType for C

FIG. 2B

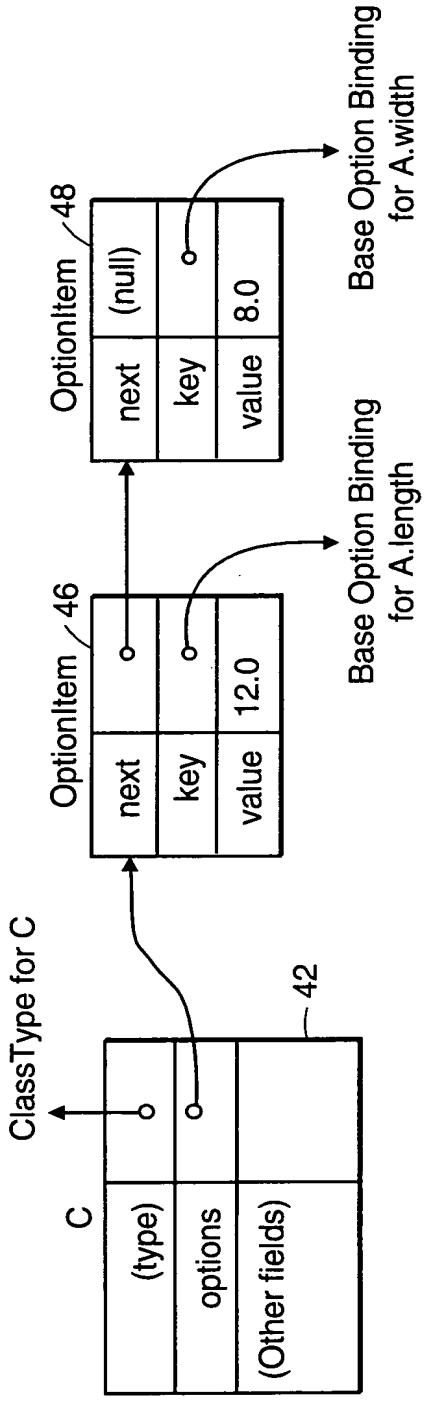
ClassType for C
C
(type)
options
(Other fields)
42
OptionItem
next
key
value
46
12.0
Base Option Binding
for A.length
OptionItem
next
key
value
48
(null)
8.0
Base Option Binding
for A.width

FIG. 2C

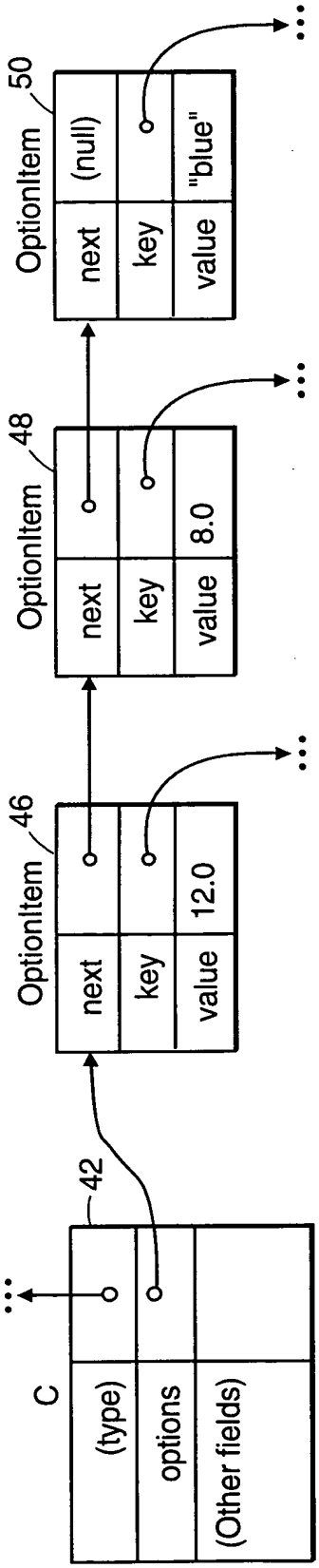

FIG. 2D

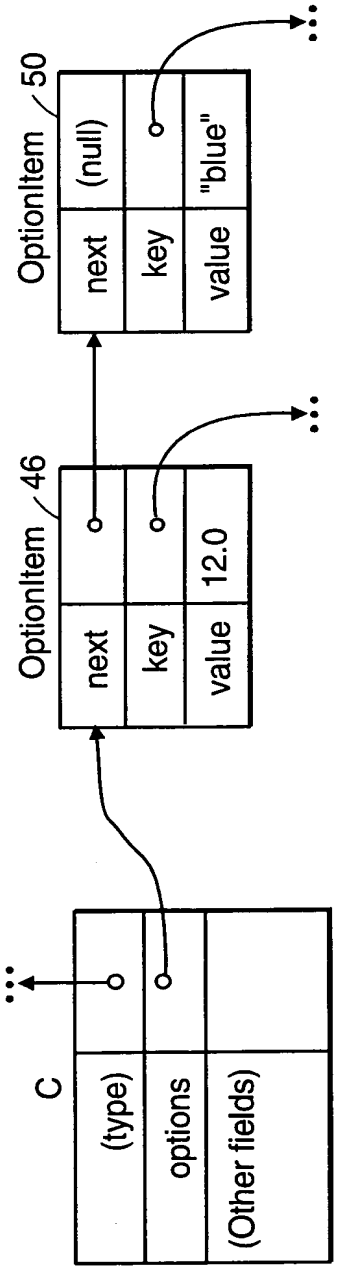

FIG. 2E

FOAFTO "20050726"

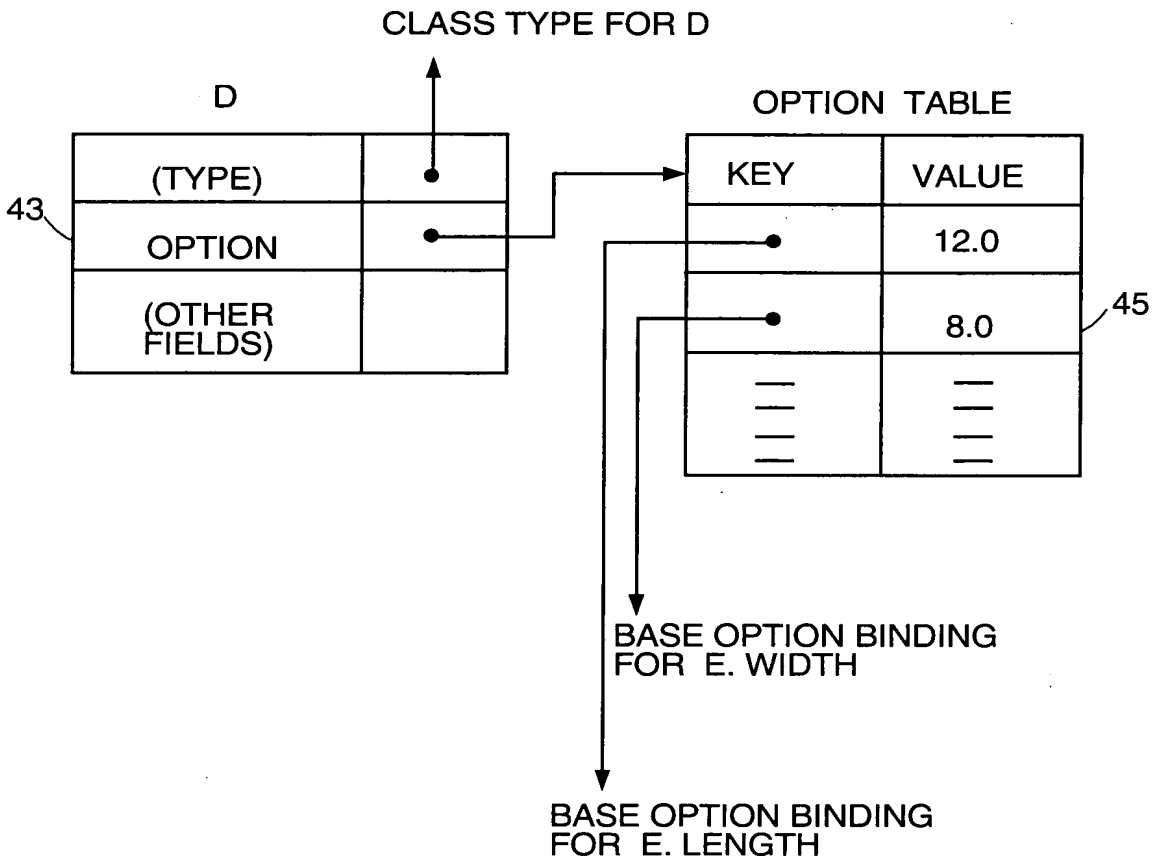

FIG. 2F

Express Mail Label No. EV 326 962 760

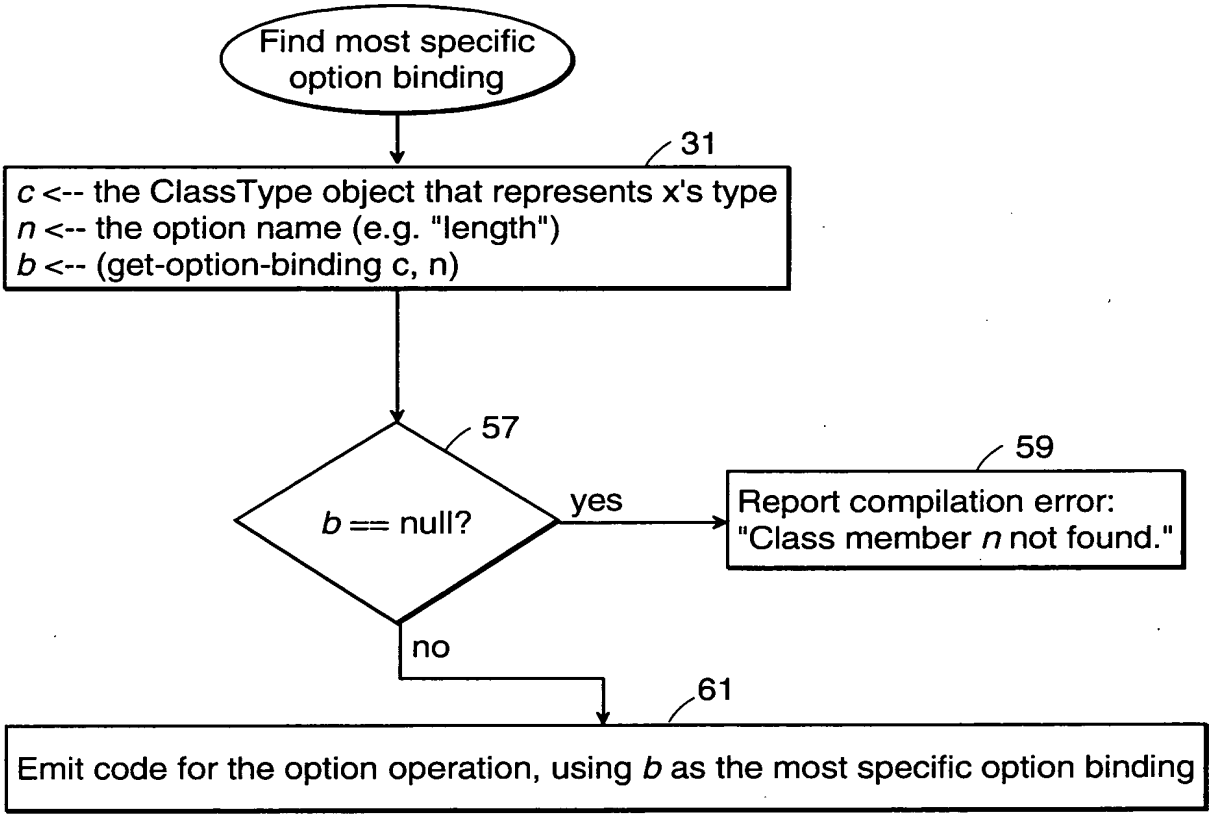

FIG. 3A

Page 26 of 260

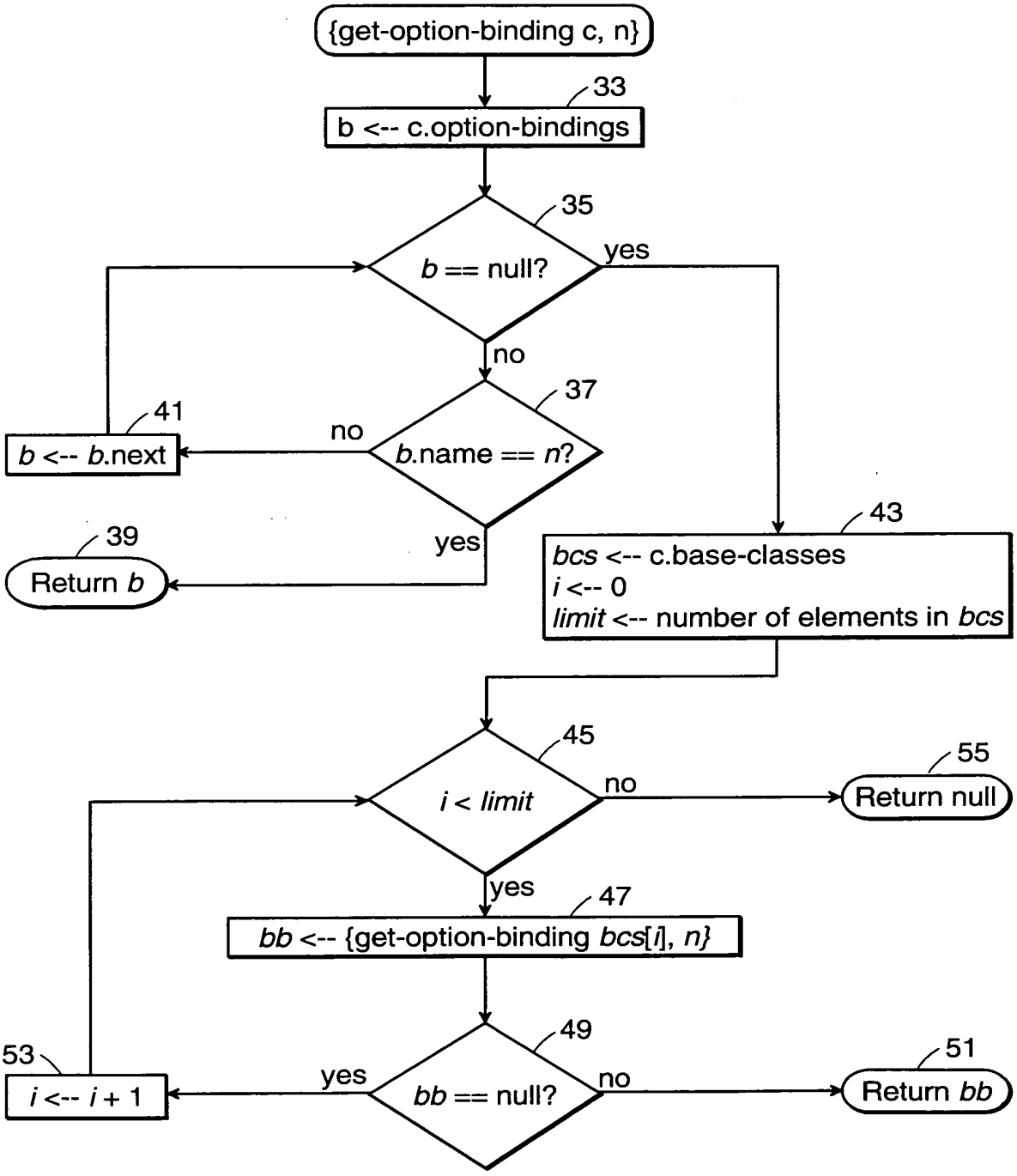
{get-option-binding c, n}
33
b <-- c.option-bindings
35
b == null?
yes
no
37
b.name == n?
no
41
b <-- b.next
yes
39
Return b
43
bcs <-- c.base-classes
i <-- 0
limit <-- number of elements in bcs
45
i < limit
no
55
Return null
yes
47
bb <-- {get-option-binding bcs[i], n}
49
bb == null?
yes
no
51
Return bb
53
i <-- i + 1

FIG. 3B

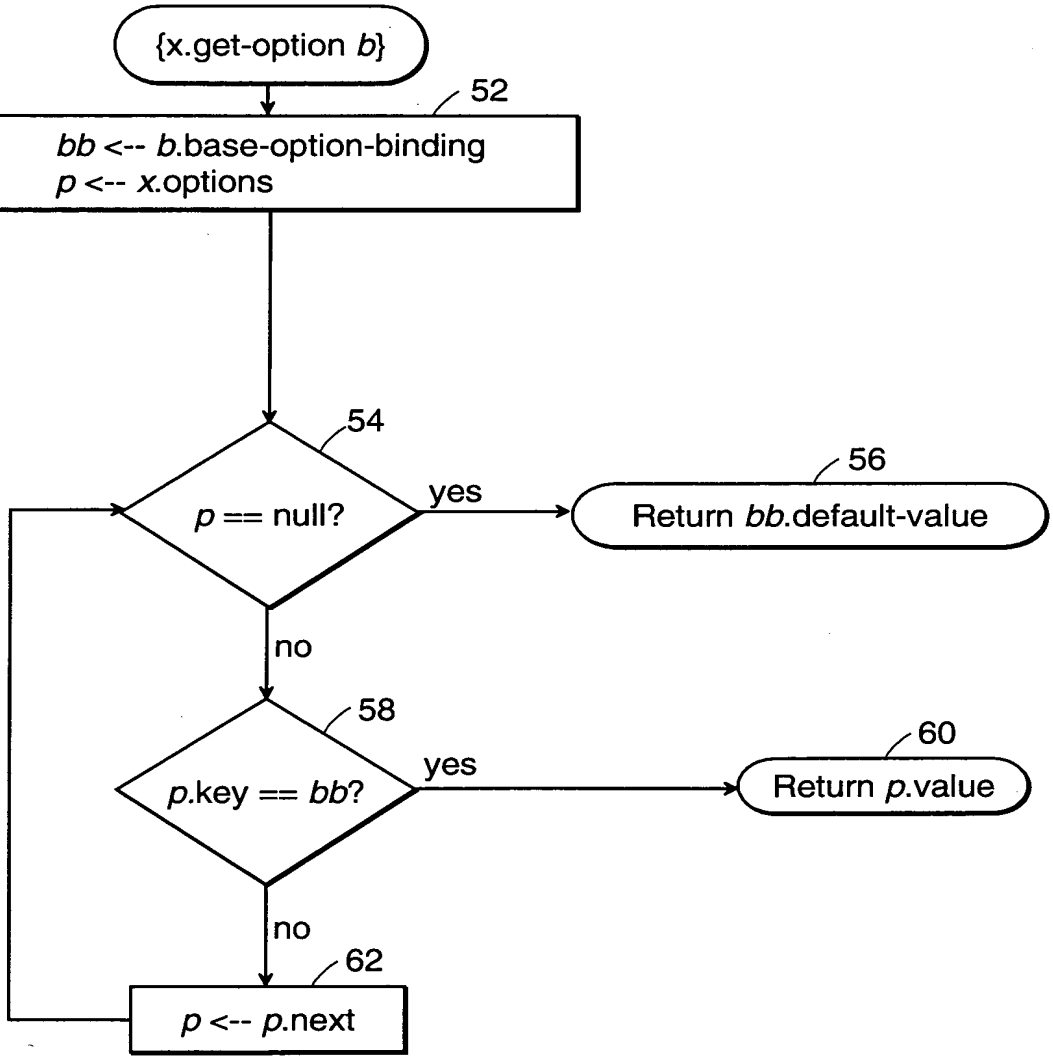
{x.get-option b}
52
bb <-- b.base-option-binding
p <-- x.options
54
p == null?
yes
56
Return bb.default-value
no
58
p.key == bb?
yes
60
Return p.value
no
62
p <-- p.next

FIG. 4

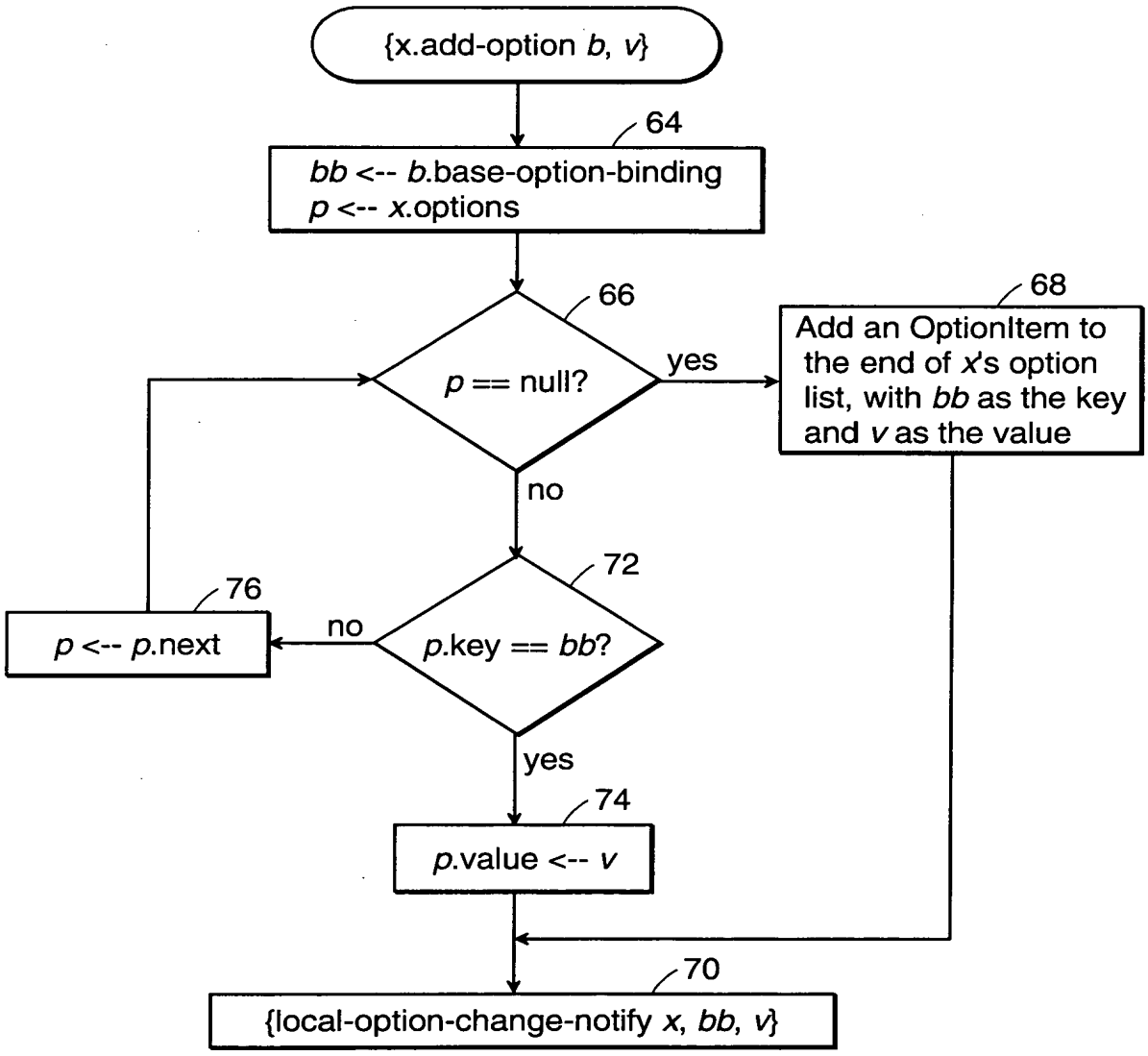
{x.add-option b, v}
64
bb <-- b.base-option-binding
p <-- x.options
66
p == null?
yes
68
Add an OptionItem to the end of x's option list, with bb as the key and v as the value
no
72
p.key == bb?
no
76
p <-- p.next
yes
74
p.value <-- v
70
{local-option-change-notify x, bb, v}

FIG. 5

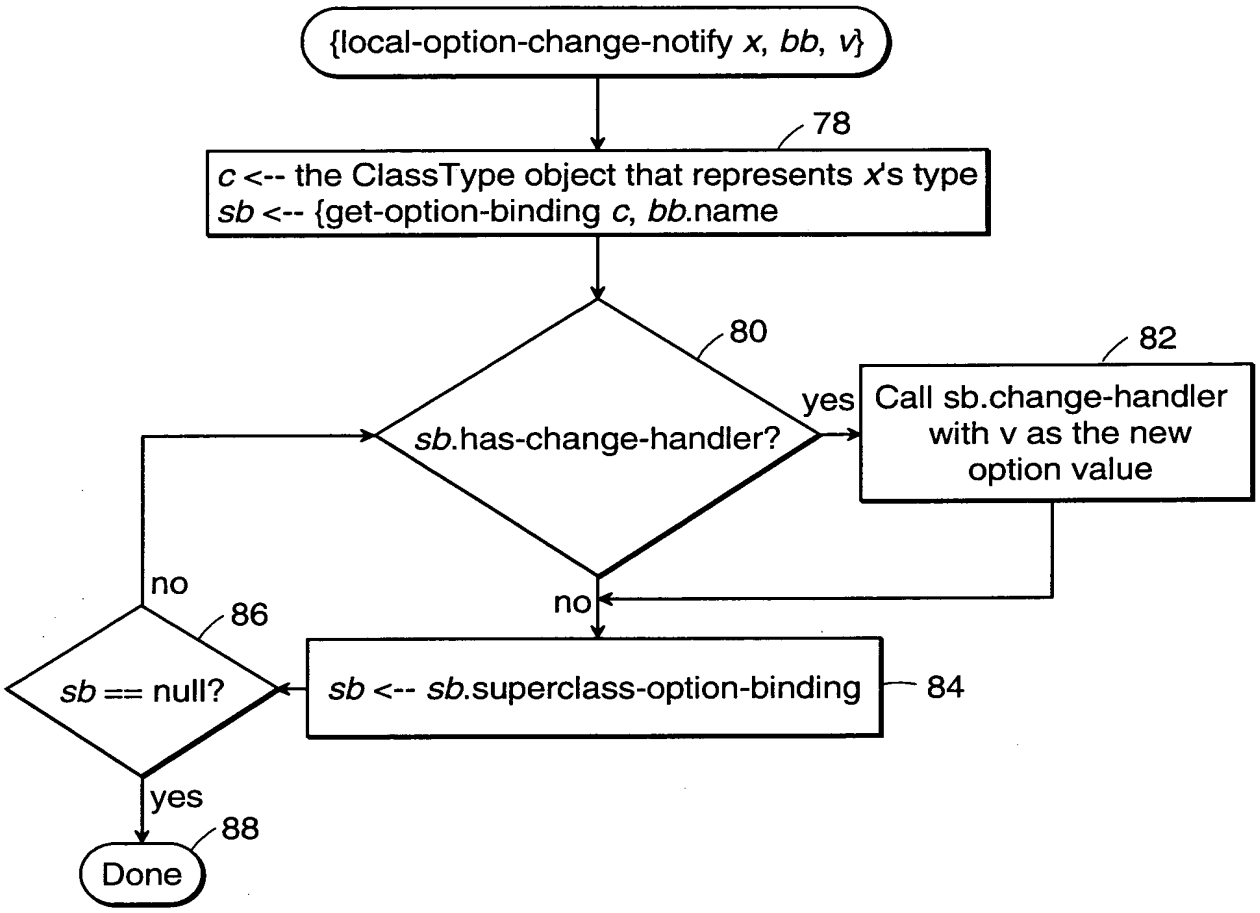
{local-option-change-notify x, bb, v}
78
c <-- the ClassType object that represents x's type
sb <-- {get-option-binding c, bb.name
80
sb.has-change-handler?
yes
82
Call sb.change-handler with v as the new option value
no
no
86
sb == null?
sb <-- sb.superclass-option-binding
84
yes
88
Done

FIG. 6

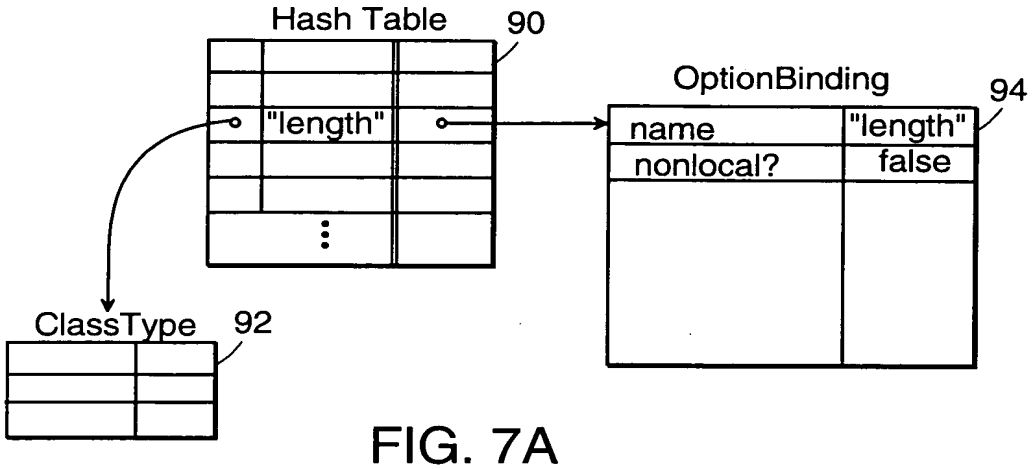

FIG. 7A

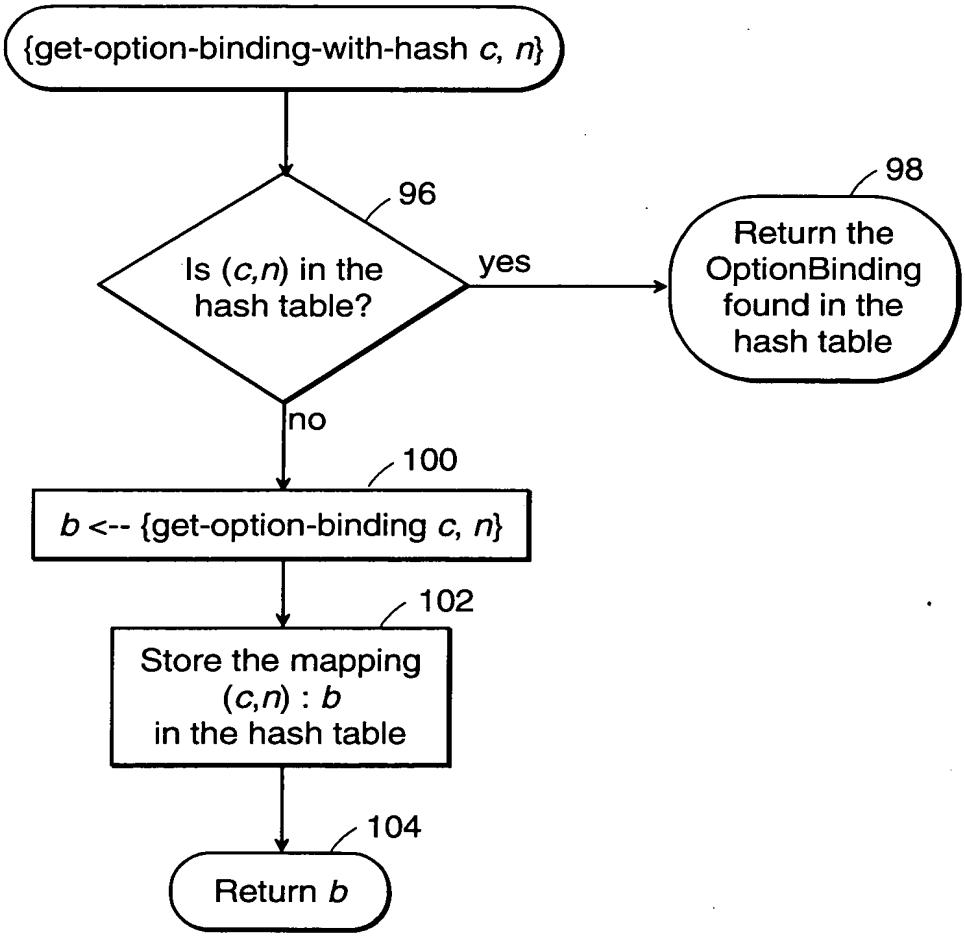

FIG. 7B

Docket No. 2682.2011-003

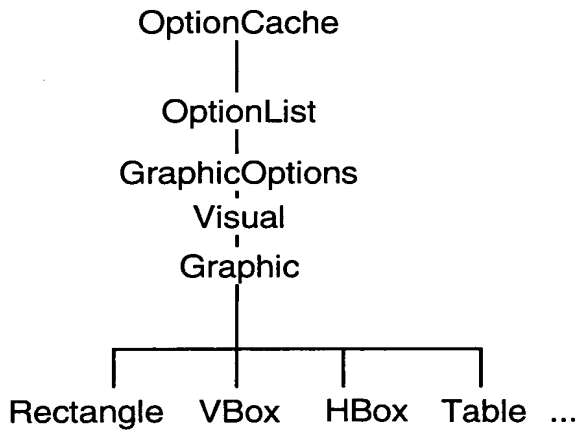
OptionCache
OptionList
GraphicOptions
Visual
Graphic
Rectangle
VBox
HBox
Table
...

FIG. 8

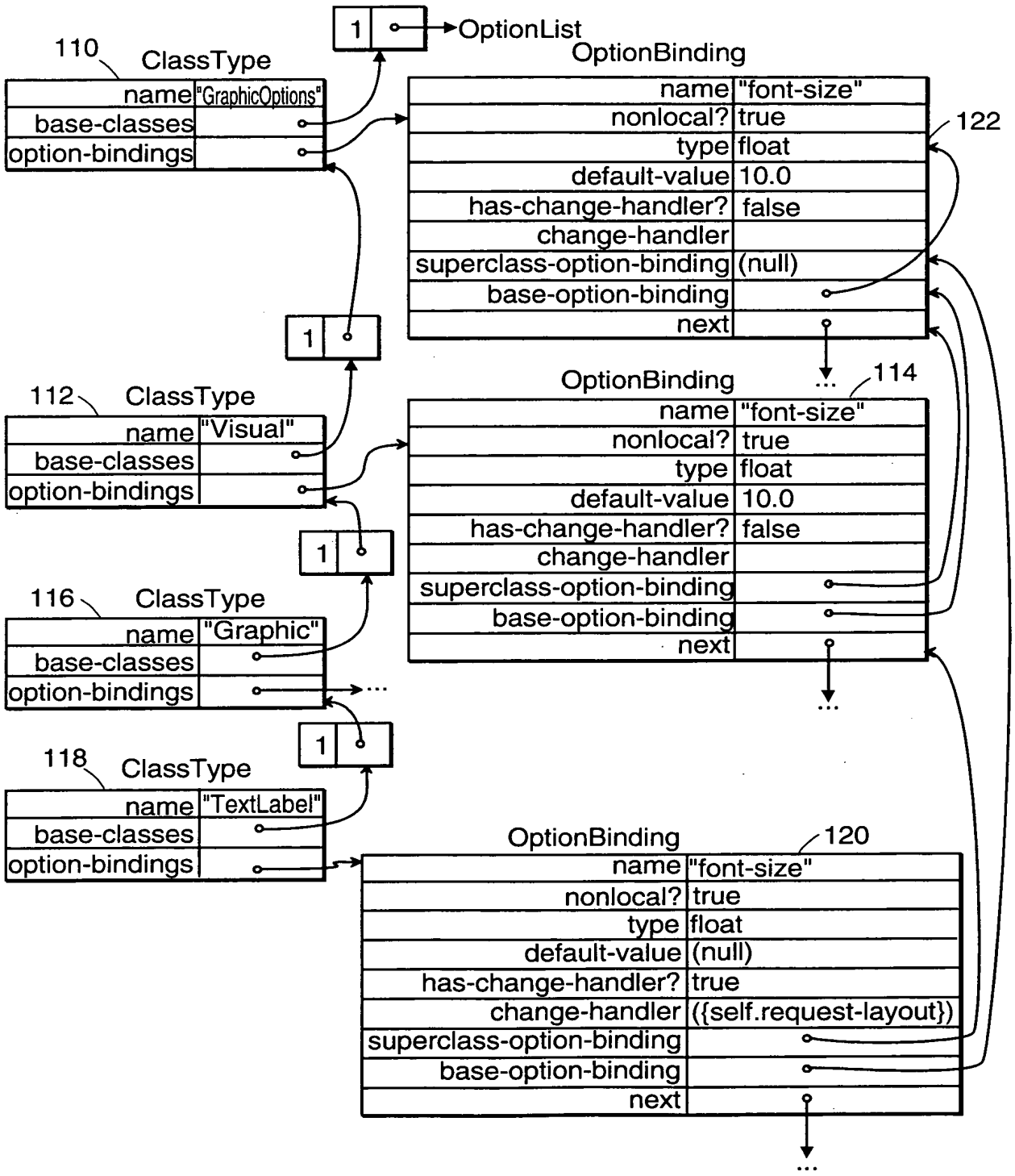
1
OptionList
110
ClassType
name "GraphicOptions"
base-classes
option-bindings
OptionBinding
name "font-size"
nonlocal? true
type float
default-value 10.0
has-change-handler? false
change-handler
superclass-option-binding (null)
base-option-binding
next
122
1
112
ClassType
name "Visual"
base-classes
option-bindings
OptionBinding
114
name "font-size"
nonlocal? true
type float
default-value 10.0
has-change-handler? false
change-handler
superclass-option-binding
base-option-binding
next
1
116
ClassType
name "Graphic"
base-classes
option-bindings
1
118
ClassType
name "TextLabel"
base-classes
option-bindings
OptionBinding
120
name "font-size"
nonlocal? true
type float
default-value (null)
has-change-handler? true
change-handler ({self.request-layout})
superclass-option-binding
base-option-binding
next

FIG. 9

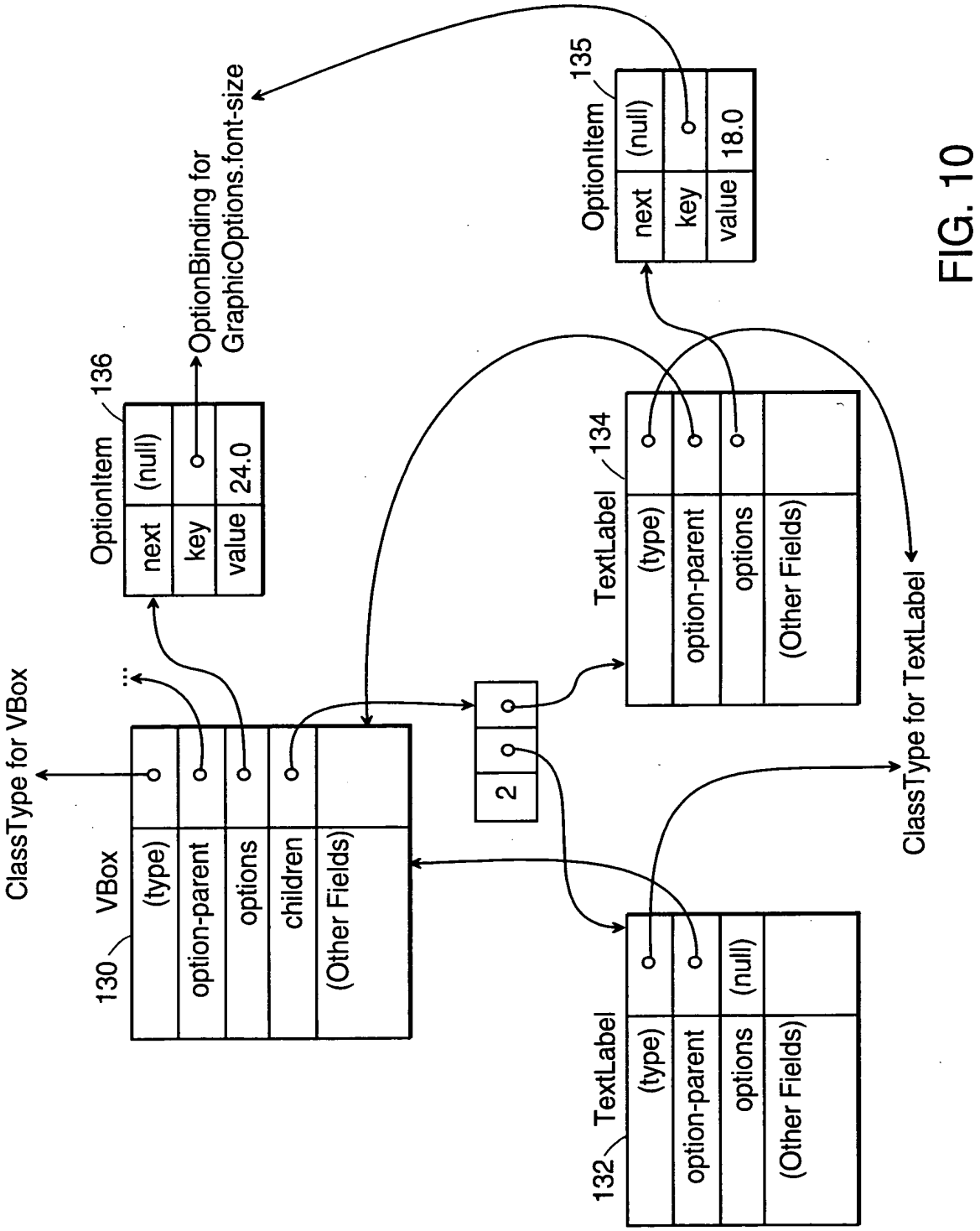

FIG. 10

EP 2 750 050 A1

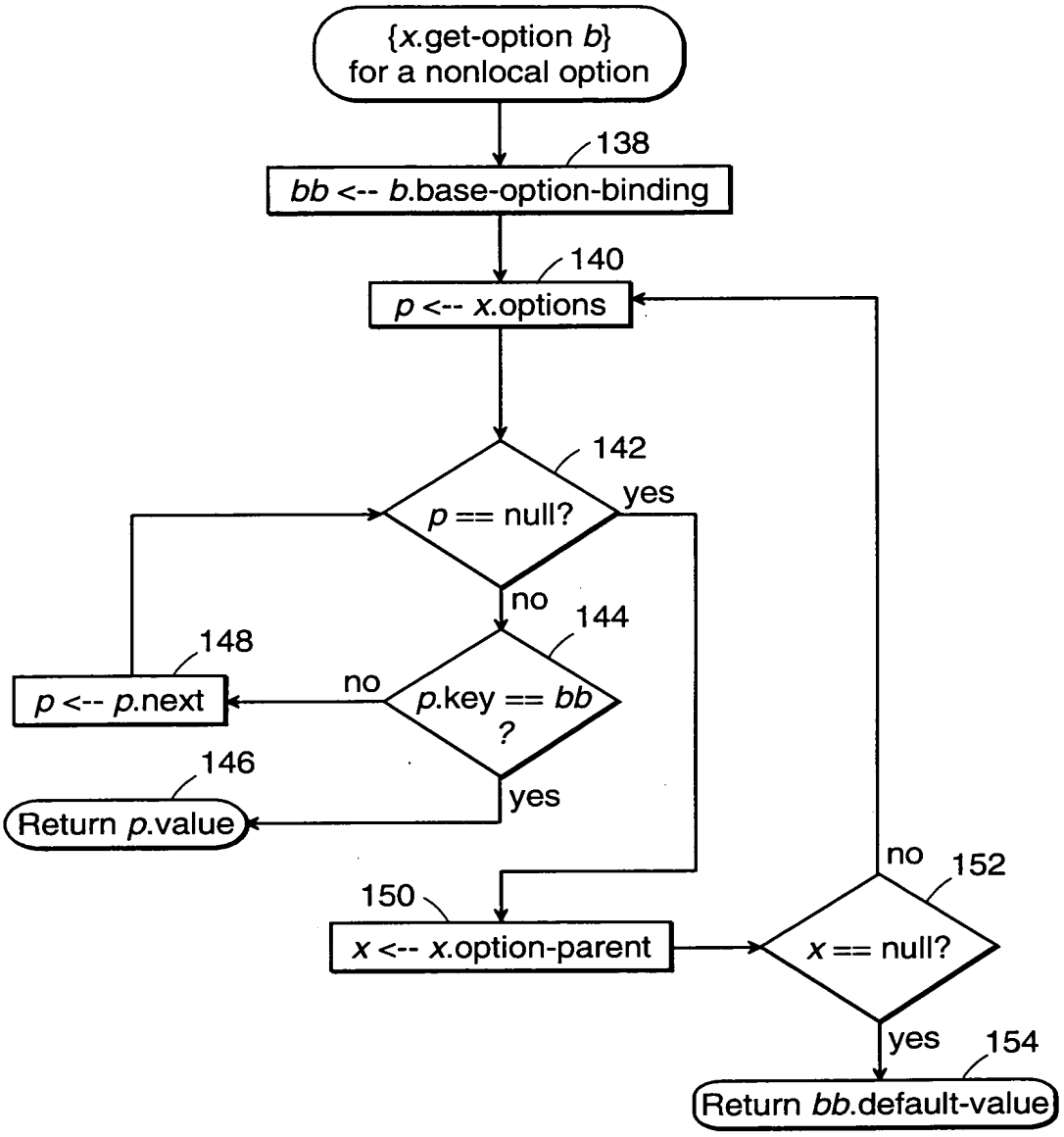

FIG. 11

Petitioner's Exhibit 1004
Page 0257

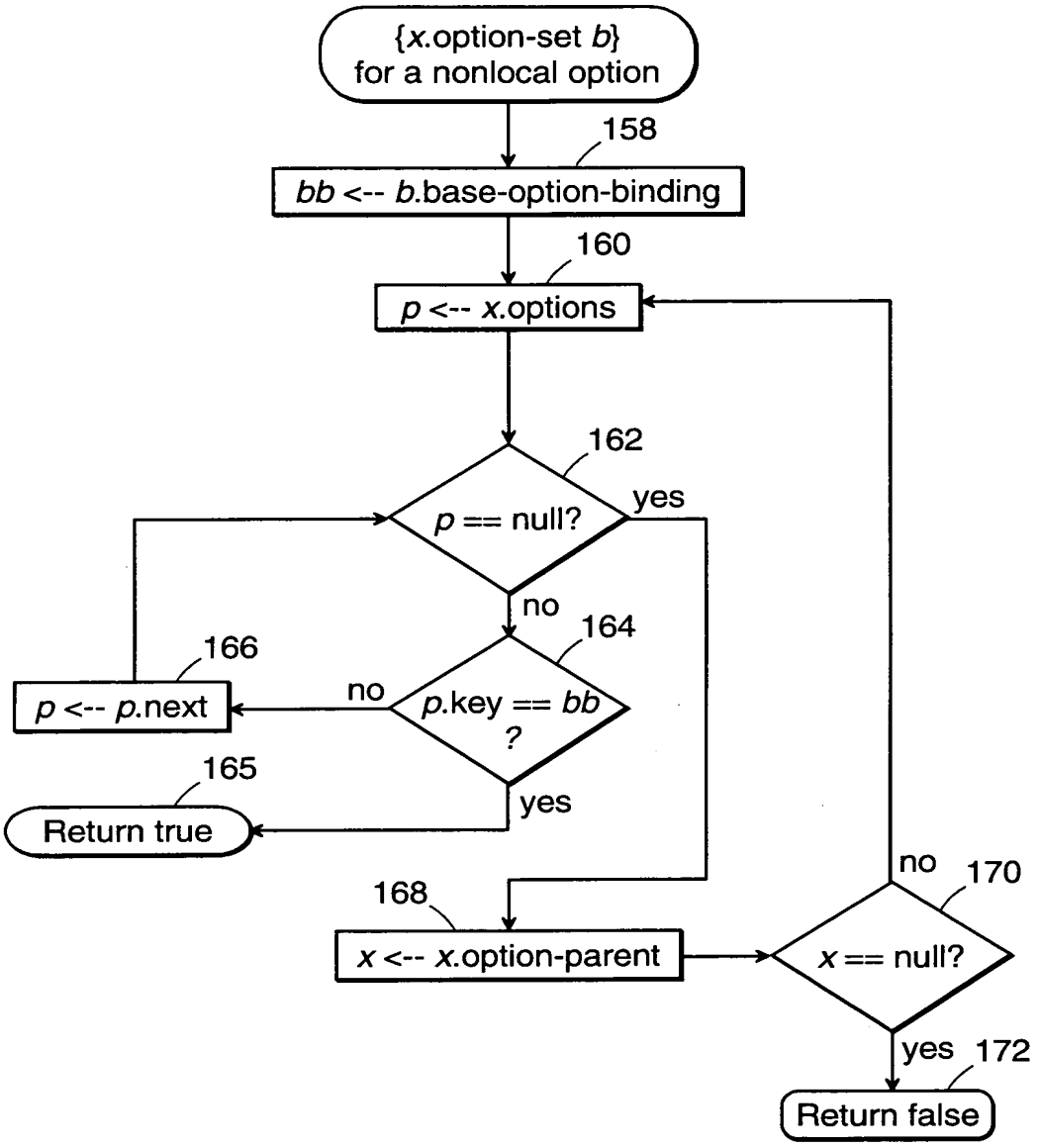
{x.option-set b}
for a nonlocal option
158
bb <-- b.base-option-binding
160
p <-- x.options
162
p == null?
yes
no
164
p.key == bb
?
no
166
p <-- p.next
yes
165
Return true
168
x <-- x.option-parent
no
170
x == null?
yes
172
Return false

FIG. 12

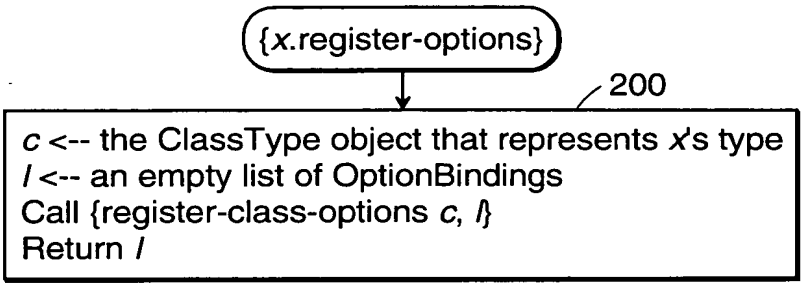

FIG. 13A

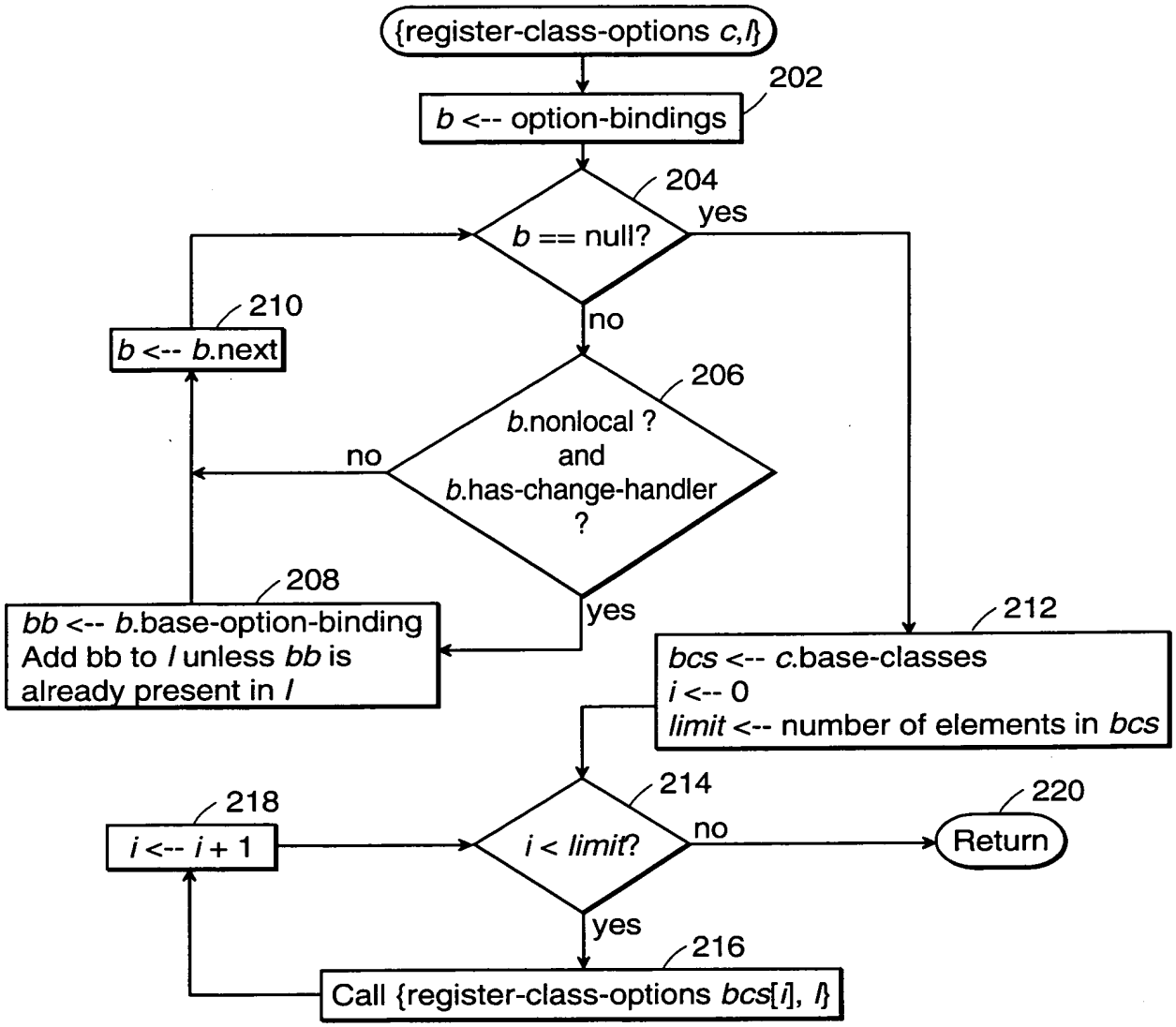

FIG. 13B

Google Exhibit 1004 - Page 0249

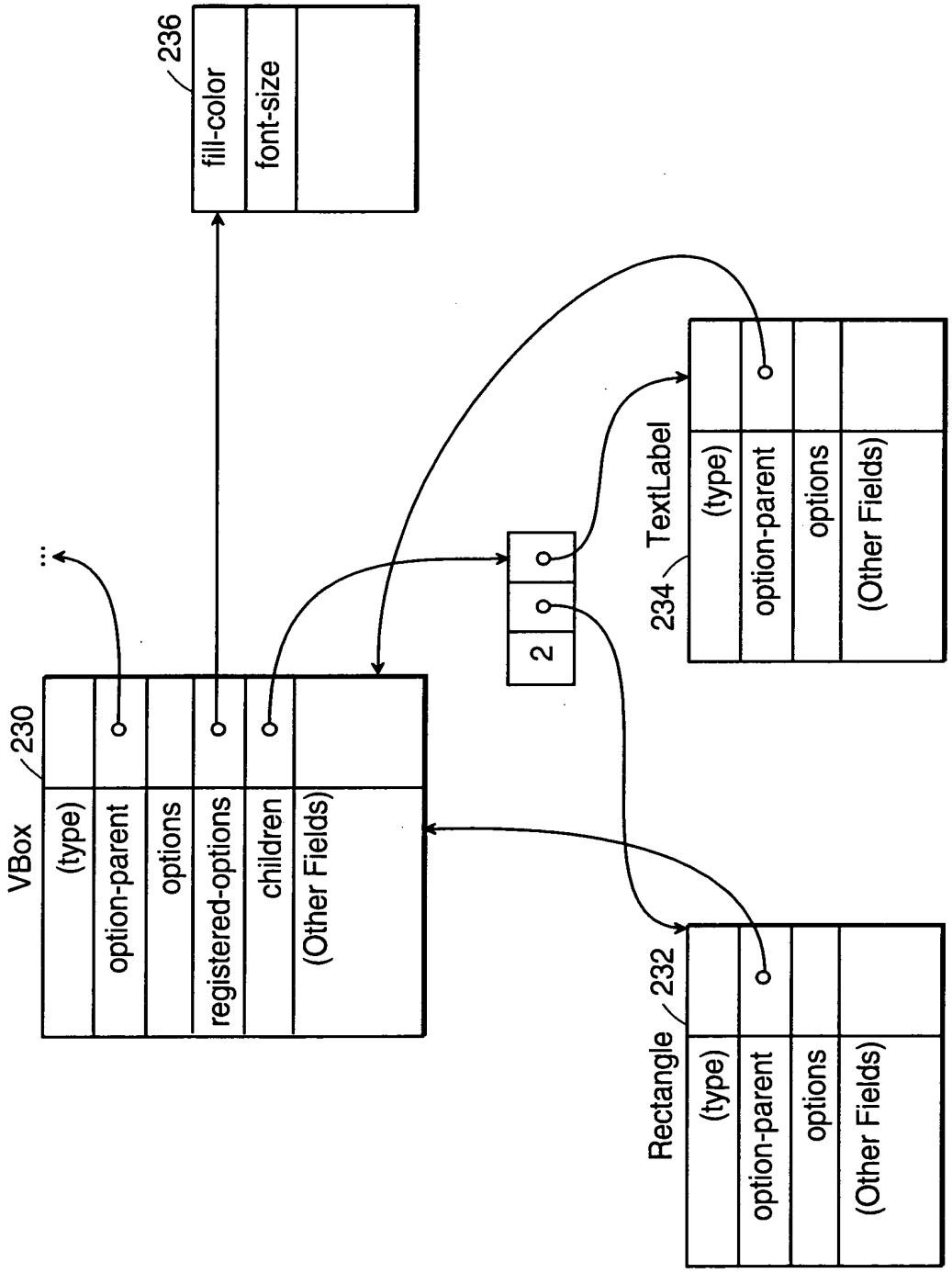
236
fill-color
font-size
VBox
230
(type)
option-parent
options
registered-options
children
(Other Fields)
2
TextLabel
234
(type)
option-parent
options
(Other Fields)
Rectangle
232
(type)
option-parent
options
(Other Fields)

FIG. 14

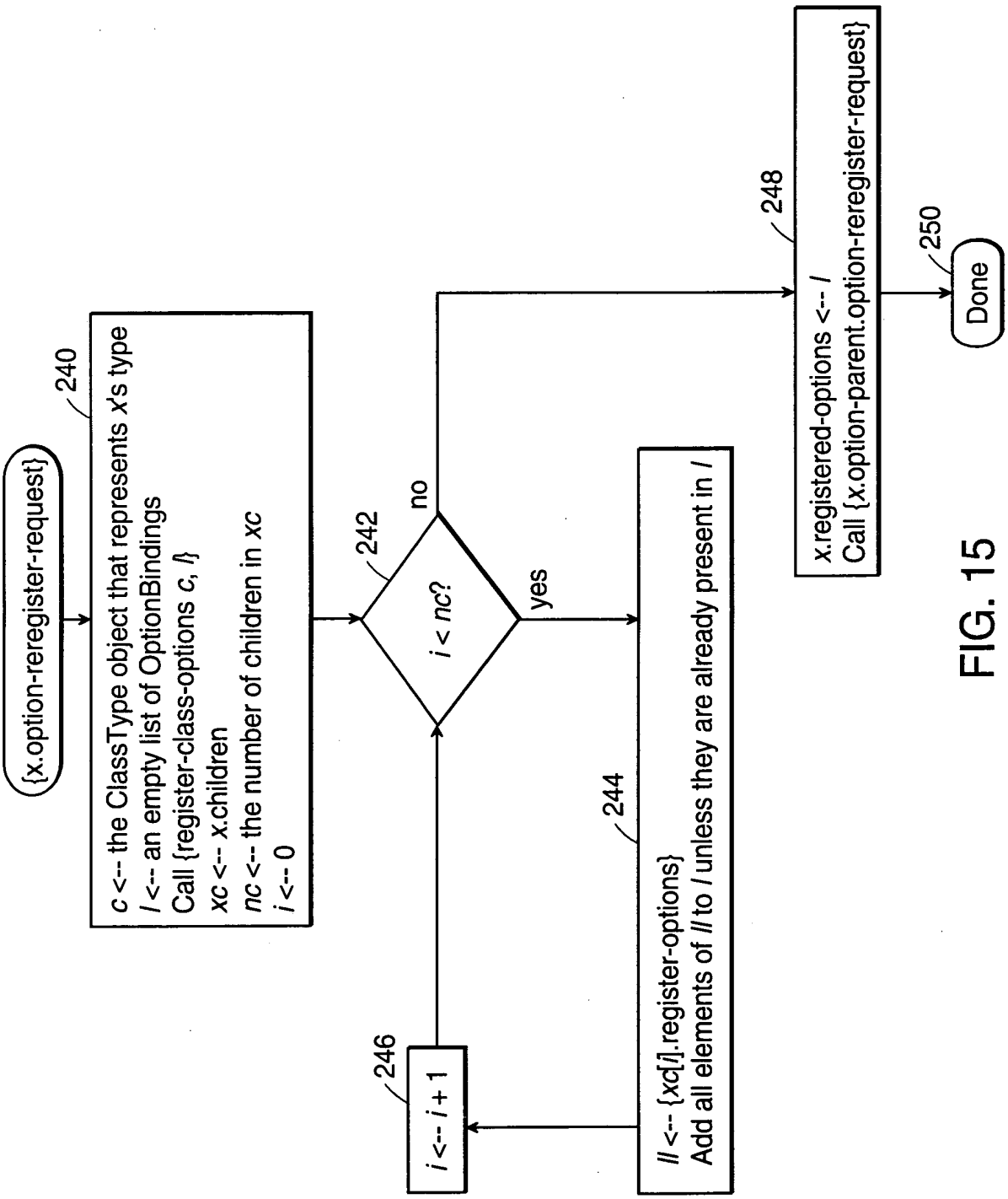
{x.option-reregister-request}
240
c <-- the ClassType object that represents x's type
l <-- an empty list of OptionBindings
Call {register-class-options c, l}
xc <-- x.children
nc <-- the number of children in xc
i <-- 0
242
i < nc?
no
yes
248
x.registered-options <-- l
Call {x.option-parent.option-reregister-request}
250
Done
244
ll <-- {xc[i].register-options}
Add all elements of ll to l unless they are already present in l
246
i <-- i + 1

FIG. 15

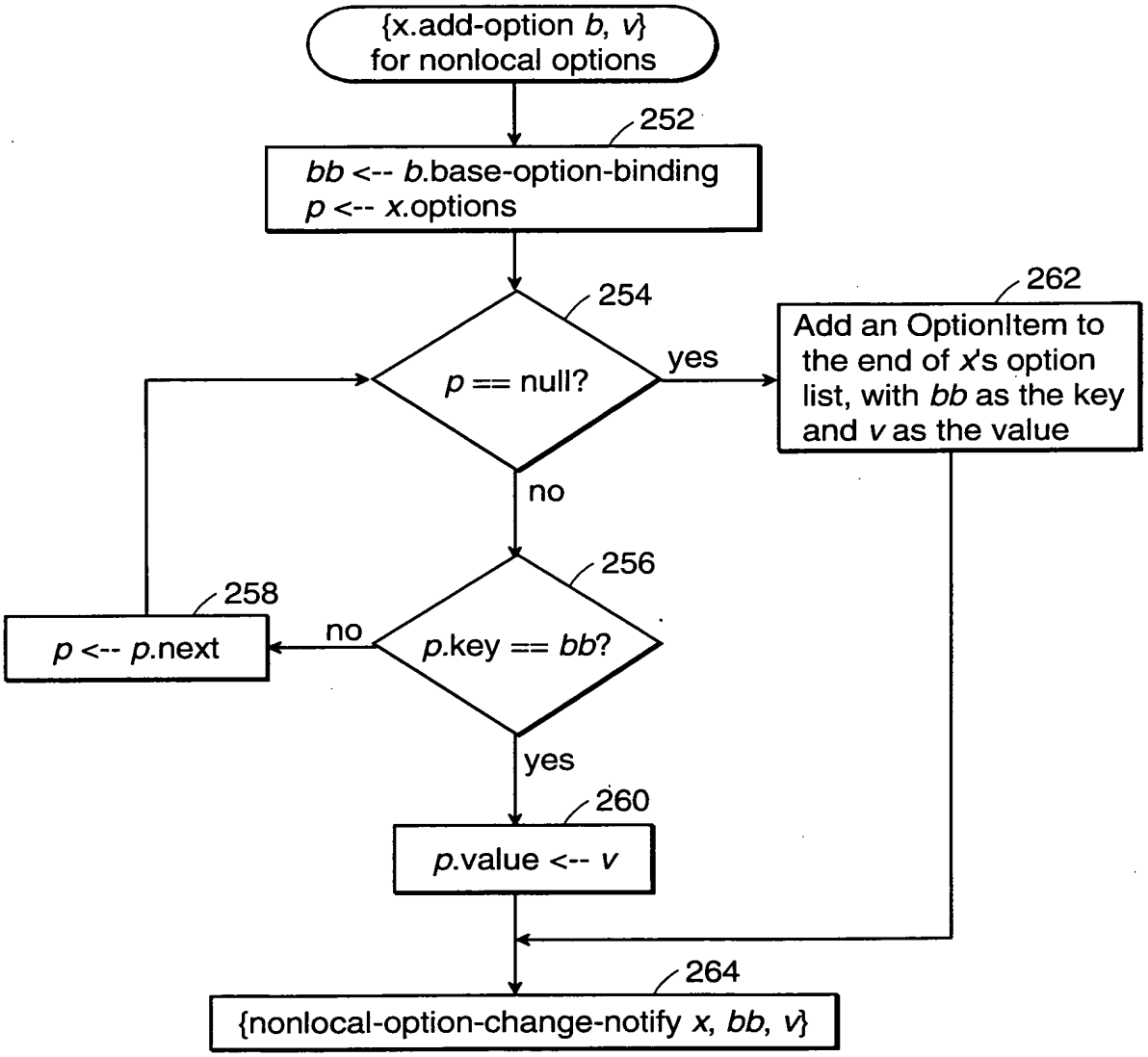
{x.add-option b, v}
for nonlocal options
252
bb <-- b.base-option-binding
p <-- x.options
254
p == null?
yes
262
Add an OptionItem to the end of x's option list, with bb as the key and v as the value
no
256
p.key == bb?
no
258
p <-- p.next
yes
260
p.value <-- v
264
{nonlocal-option-change-notify x, bb, v}

FIG. 16

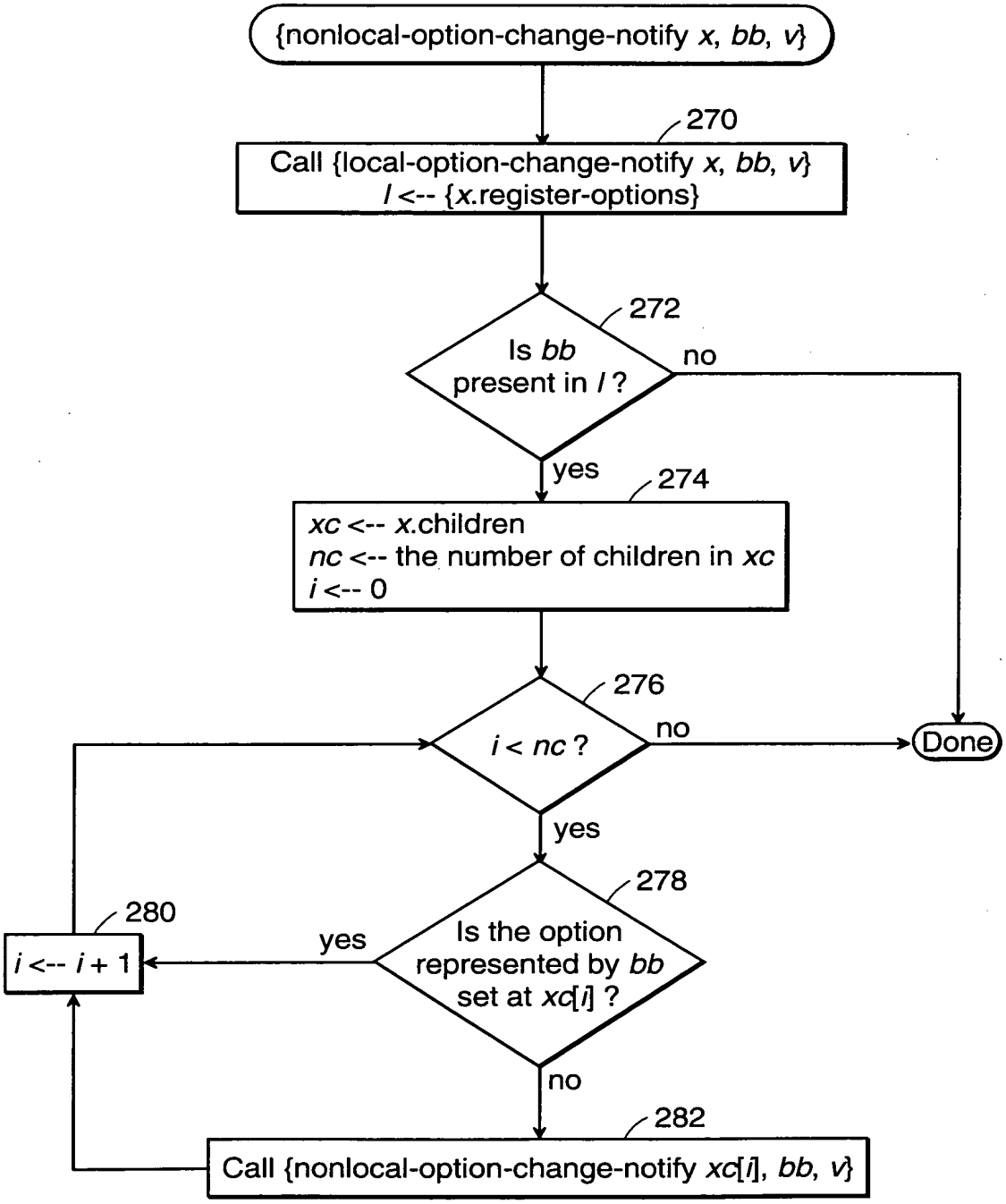
{nonlocal-option-change-notify x, bb, v}
270
Call {local-option-change-notify x, bb, v}
l <-- {x.register-options}
272
Is bb present in l ?
no
yes
274
xc <-- x.children
nc <-- the number of children in xc
i <-- 0
276
i < nc ?
no
Done
yes
278
Is the option represented by bb set at xc[i] ?
yes
280
i <-- i + 1
no
282
Call {nonlocal-option-change-notify xc[i], bb, v}

FIG. 17

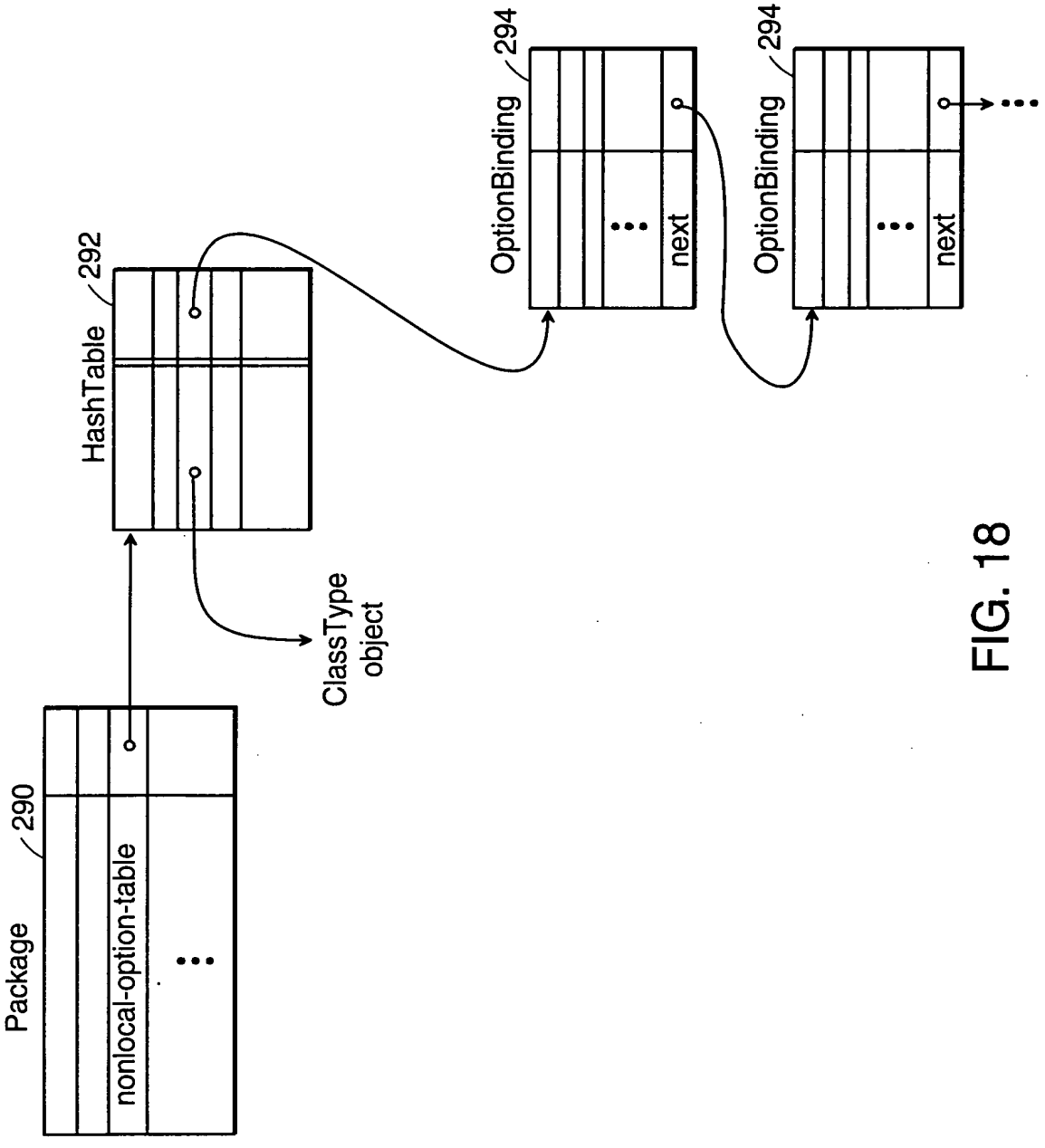
Package
290
nonlocal-option-table
ClassType object
HashTable
292
OptionBinding
294
next
OptionBinding
294
next

FIG. 18

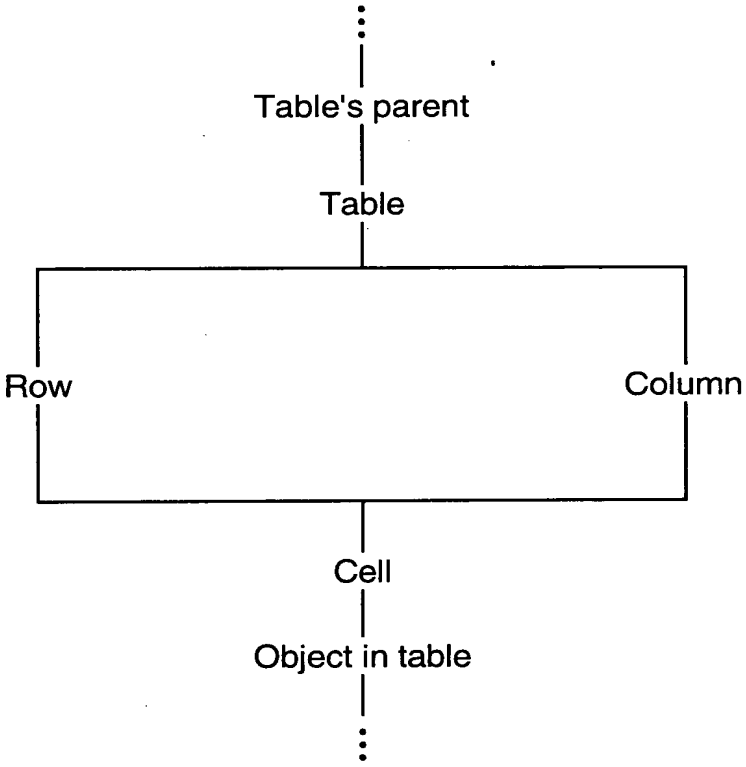

FIG. 19

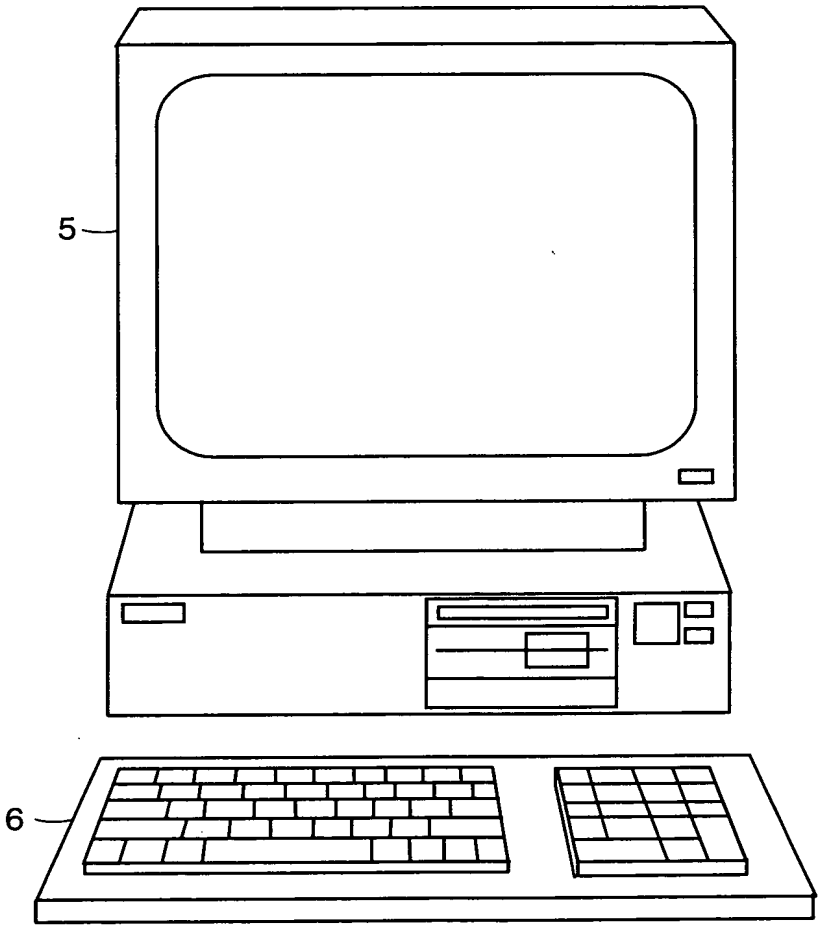

FIG. 20

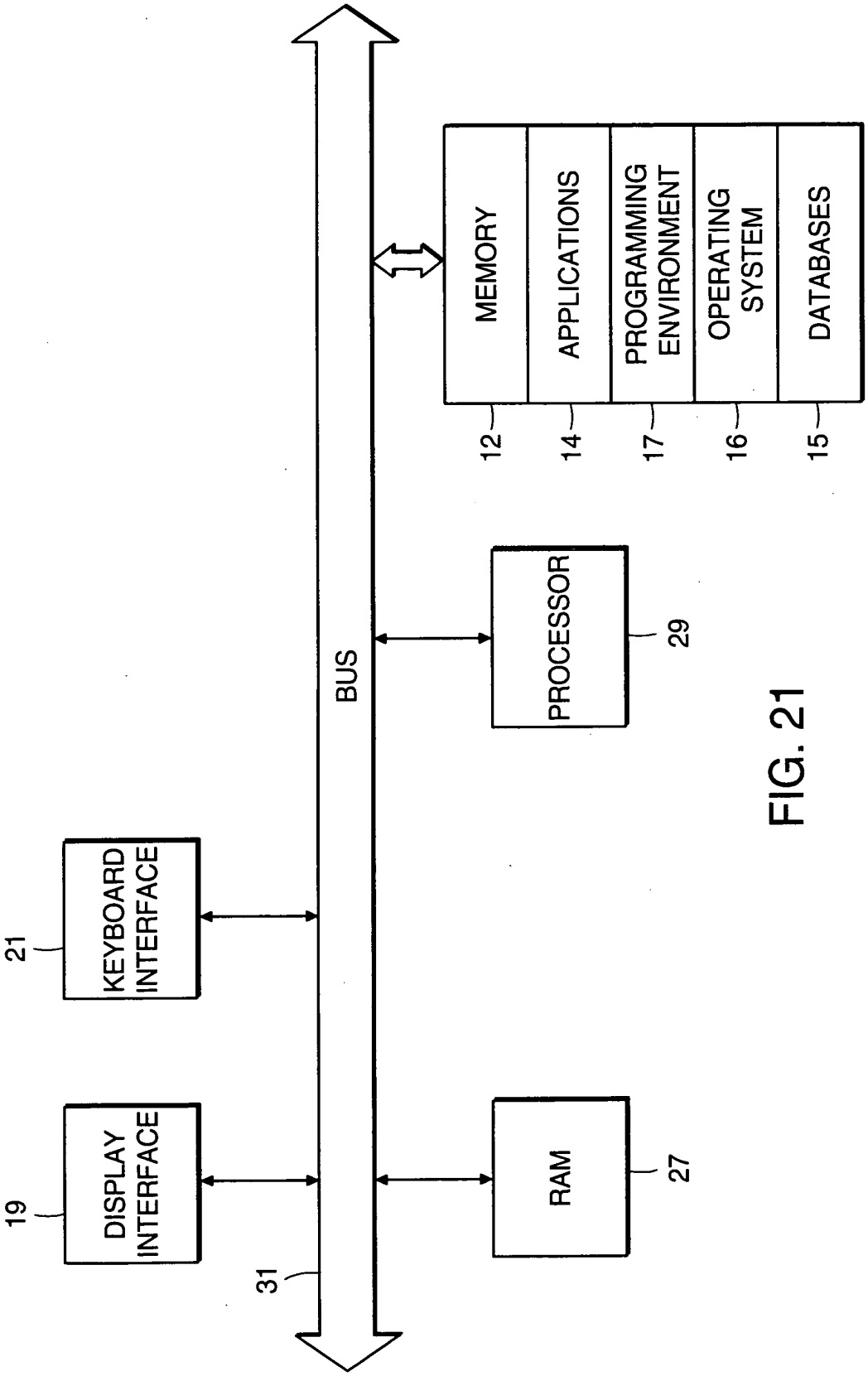

FIG. 21

TO2TP0 2565760